

## REMARKS

The above amendment and these remarks responsive to the Office Action of Examiner Anita Choudhary, dated 12 Feb 2004.

Claims 1-19 are in the case, none as yet allowed.

### 35 U.S.C. 103

Claims 1-19 have been rejected under 35 U.S.C. 103(a) over U.S. Patent 6,473,800 (Jerger) in view of U.S. Patent 5,974,549 (Golan).

Jerger describes an ActiveX construct, and thus may be construed to provide to the user the ability to visit a web site and to accept a download from that site. This is not the same as having a user visit a site and grant to that site the ability to take over, in the sense of managing policies, on the user's terminal.

Moriconi relates to central access control by a server. Rules established by the server administrator are pushed out to the terminals defining who can run what.

Together these references do not teach nor render

obvious applicants' invention, which provides to the user the ability to visit many web sites and choose to grant or deny to a visited site the ability to, in effect, take over the user's terminal - in the sense of not only downloading files or applications but also to manage security policies on that user terminal.

On this point, of combining the references, the Examiner states (for example, with respect to claim 1):

"Given these features... modifying the system shown by Jerger to employing the well known features shown by Moriconi, in order to more securely administer the distribution of sensitive application information from a secure server site to a client (see Moriconi col. 3 lines 32-48)." [Office Action, page 4.]

Jerger here teaches that an enterprise may distribute an authorization infrastructure to allow the enterprise to protect its assets against unauthorized access.

On the other hand, applicants' invention relates to a user client allowing a trusted site to manage its (the client's) policies.

Consequently, applicants traverse the rejection of claims 1-19 under 35 U.S.C. 103(a), arguing that the Examiner has not made the required prima facie case of obvious under the statute.

In applicants' invention, the administrator, not the end user, determines what the code gets to do. And, it is the user that grants that authority or control to the server (administrator). Applicants' claimed trust model inherently is about the end user deciding to trust the administrator (that is, server), not the code-author (as is the case with Jerger) to determine the rights-level of the code executing on the client machine.

In applicants' invention, the administrator, once granted the authority by the user, can subsequently turn up/down the rights level of the code. Basically, applicants' model is a security partitioning for "grid-style" processing (a term which has been added to the art subsequent to applicants' filing, but refers to the concept applicants claim). In such "grid-style" processing, a secured portion of the end-user's machine is "borrowed" to run whatever the website, or server, administrator wants to run. But, in accordance with applicants' invention, it is

the user that grants that control to the administrator.

In applicants invention, the user determines if the site (server, administrator) is to be trusted, and if trusted, the administrator, or server, controls what is to be loaded to and executed at the client. See applicants' specification, page 4, lines 6-19; page 103, line 12 to page 105, line 8, which teach:

"Known in the art are systems and methods for downloading custom code from a web site to a client machine. Some download models provide no way for the user to grant permission, others provide the user with the opportunity to grant permission, usually based upon trust in the provider (author) of the code: that is, does the prospective user believe the code is attributed to the correct vendor, and does the prospective user trust that vendor to load code to his, the client, machine. There is a need in the art for a system and method for conditioning download authorization on web site verification, as distinguished from code vendor identification, in order to provide an improved accountability chain with respect to code downloaded from the web."

[Specification, page 4, lines 6-19.]

Applicants previously amended each of the independent claims to make clear that the administrator (server), once accepted as trusted by the client, controls what services are executed at the client.

In their specification, applicants describe this aspect of their invention with respect to Figures 13 and 26. First, the user is queried to determine if the site is to be trusted. This is described as follows:

"...to execute custom code install from the server to the client. This involves the creation of a permission moment, a moment in which the user is prompted to respond to two queries: (1) site identity: does the user believe that the server is who it represents itself to be; and (2) site trust: does the user trust the server to place the custom code on the client machine."

"Referring to Figure 26, in accordance with the preferred embodiment of the invention, site identity is

associated with the secure sockets (SSL) signature, and whether the connection to the web site has been made using HTTPS (secure) or HTTP (not secure). If in step 553 it is determined that the user has connected to the server web site in step 551 using SSL, then the site identity and site trust queries are presented by stating (1) in step 555, the site has been verified as being what it represents itself to be, and (2) in step 556, asking "Do you trust the web site to download custom code to your client machine?" If the user has not connected to the server web site using SSL, then the site identity and site trust queries are presented by stating (1) in step 554, the site has not been verified as being what it represents itself to be, and (2) again in step 556, asking if the web site is to be trusted to download custom code to the client machine?" [Specification, page 133, line 7 to page 134, line 8. Emphasis added.]

Thereafter, code is downloaded from the server based upon user acceptance of the web site. This is described as follows:

"Download of the custom code proceeds based upon the

user determination in step 557 that the web site,  
whether verified or not, is to be trusted.

[Specification, page 134, lines 9-11. Emphasis added.]

As noted above, once the user accepts downloads, from that point on things are pushed down from the server. This is further illustrated in Fig. 13, which shows how cross certificates 566, restricted list 570, unrestricted lists 573 are pushed down from server 100 to client 200 to affect what runs in a subscription 201 -- these operations occurring well after the user 200 has accepted site 100 as a trusted site.

Applicants have amended the independent claims 1, 12, 18 and 19 to make clear that only once a site is accepted as trusted by the user, security policies are centrally administered by the system administrator.

The Jerger model does not allow for multiple versions of an ActiveX to exist simultaneously on a client machine, each executing in wholly separate execution spaces under different security policies, all unaware of each other and non-conflicting.

On the other hand, applicants' model does keep the security context of downloaded active content in completely separate spaces, governed only by policies that come from the separates source sites: the sites are fundamentally unable to affect each other's components even if they are literally the same components.

For example, if two different travel websites utilize and ActiveX signed by Mapquest according to Verisign, if travel site A upgrades ists ActiveX, then end users that use both sites would be forced into using the most recent version, even if this might break travel site B. Nor does one travel site have the ability to protect its use of the ActiveX only to its own context by setting policies on it and blocking it from third-party use.

With applicants' model, both travel sites could have the same (or even same with version deltas) signed agent that came from Mapquest, and be guaranteed that those agents would run in private non-interacting secure spaces on the end user machine, spaces that can even be encrypted.

This aspect of applicants' invention may be illustrated with reference to Figure 13. In Figure 13, signed agents

562, 563 can be the exact same agent code (for example, "out of office agent, published/signed by IBM"), but of slightly different versions (e.g. 1.1 vs 1.2). Figure 13 shows that client 200 has separate execution spaces for subscriptions 202 and 201, with separate agents 562, 563 within each subscription. Indeed, even the policy is separate... that is, server B 101 could choose, for example, to move this agent 561 from its unrestricted list 574 to its restricted list 571, and that would only impact its offline version. Server B 101 has no way of affecting the files from server A 100, or even being aware of them. Figure 13 is described in applicants' specification as follows:

" Referring to Figure 13, system components exercised in qualifying signed agents 560, 561 from a plurality of servers 100, 101 for execution as signed agents 562, 563 at a client 200 are illustrated. Server directors 350, 351 include certificates 564, 565, cross certificates 566, 567, downloadable cross certificates 568, 569, restricted group lists 570, 571 and unrestricted group lists 573, 574. Client 200 includes client side rendition 202, 562 of application 136 with signed agent(s) 560, and client side rendition 201, 563 of server application 137 with signed agent(s)

561; and client directory 212 with downloadable cross certificates 576, unionized restricted group list 572 and unionized unrestricted group list 575. A signature is a name plus an electronic certificate. Group lists 572, 575 include names, not complete signatures. Unionized group lists include the union of names 570, 571 and 573, 574 from all servers, in this example two servers 100 and 101 are shown, but there may be more." [Specification, page 44, line 11 to page 45, line 6. Emphasis added.]

As will be apparent to those of skill in the art, it is inherent that unionized group lists 572, 575 imply that servers A and B are unaware of each other, and to do a union, agents 562 and 563 must run in separate partitions.

There is nothing in the Jerger (Microsoft) model that allows off-line enabled, completely partitioned execution spaces, where the security policy of what happens in those spaces can be centrally administered. Microsoft JVMs and applets are the closest thing, in that applets from server A vs server B can run in partitions, but these have no offline-ability built into them. As such, the applets will run when the user is connected, but the site has no

meaningful generic way to push its data to a desktop, along with processing logic and a snapshot of security policies so that this will all run seamlessly the same way when the user is disconnected.

Applicants previously amended the independent claims 1, 12, 18 and 19 to clarify that simultaneously existing agents execute in separate, non-conflicting execution spaces.

The Examiner refers to Moriconi, col. 8, lines 7-14 and col. 10, lines 27-32, stating that such teaches "Download utilities for downloading services and programs from server site to separate and non-conflicting execution spaces at said users client..."

This is what Moriconi teaches:

"Role membership may be further constrained by the notion of mutual exclusion. Two roles are mutually exclusive if no single user can be granted to both roles simultaneously. Role mutual exclusion implements a business requirement of separation of duty. For example, a submit\_budget role and an approve\_budget role should be mutually exclusive, because no user

should be simultaneously authorized to perform both actions." (Moriconi, col. 8, lines 7-14).

"Referring now to FIG. 5, a block diagram of one embodiment of application guard 310, located within non-volatile memory 138 in FIG. 3, is shown.

Application guard 310 may be distributed to clients 116 throughout an enterprise, and is designed to reside along with each of the protected applications having an associated application guard 310." (Moriconi, col. 10, lines 27-32.)

Here the Examiner is apparently applying the concept of mutually exclusive roles from Moriconi to applicants' concept of non-conflicting execution spaces, as discussed above. Applicants' argue that these are different concepts, and that is not proper to cite Moriconi for this limitation.

Applicants request that the amendment to claim 1 be entered, and claims 1-19 be allowed.

### **SUMMARY AND CONCLUSION**

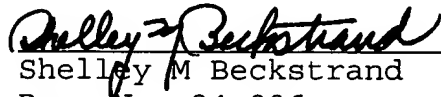
Applicants urge that the above amendments be entered and the case passed to issue with claims 1-19.

The Application is believed to be in condition for allowance and such action by the Examiner is urged. Should differences remain, however, which do not place one/more of the remaining claims in condition for allowance, the Examiner is requested to phone the undersigned at the number provided below for the purpose of providing constructive assistance and suggestions in accordance with M.P.E.P. Sections 707.02(j) and 707.03 in order that allowable claims can be presented, thereby placing the Application in condition for allowance without further proceedings being necessary.

Sincerely,

Carl J. Kraenzel, et al.

By

  
Shelley M Beckstrand  
Reg. No. 24,886

Date: 12 July 2004

Shelley M Beckstrand, P.C.  
Attorney at Law  
314 Main Street  
Owego, NY 13827

Phone: (607) 687-9913  
Fax: (607) 687-7848